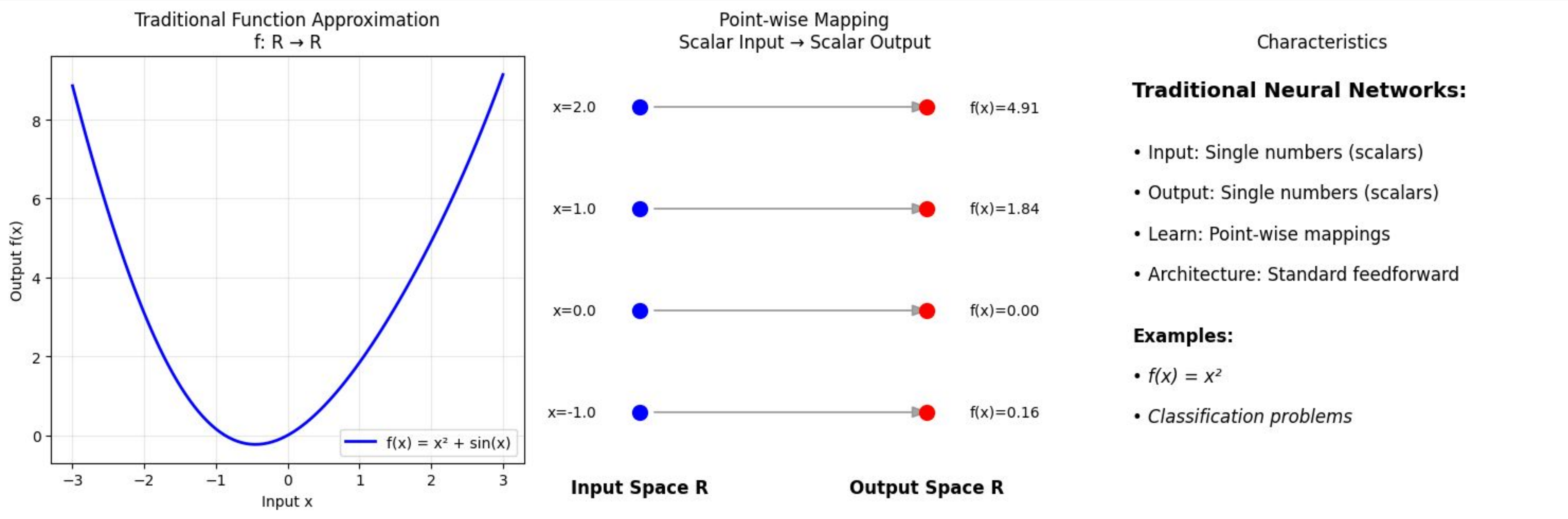
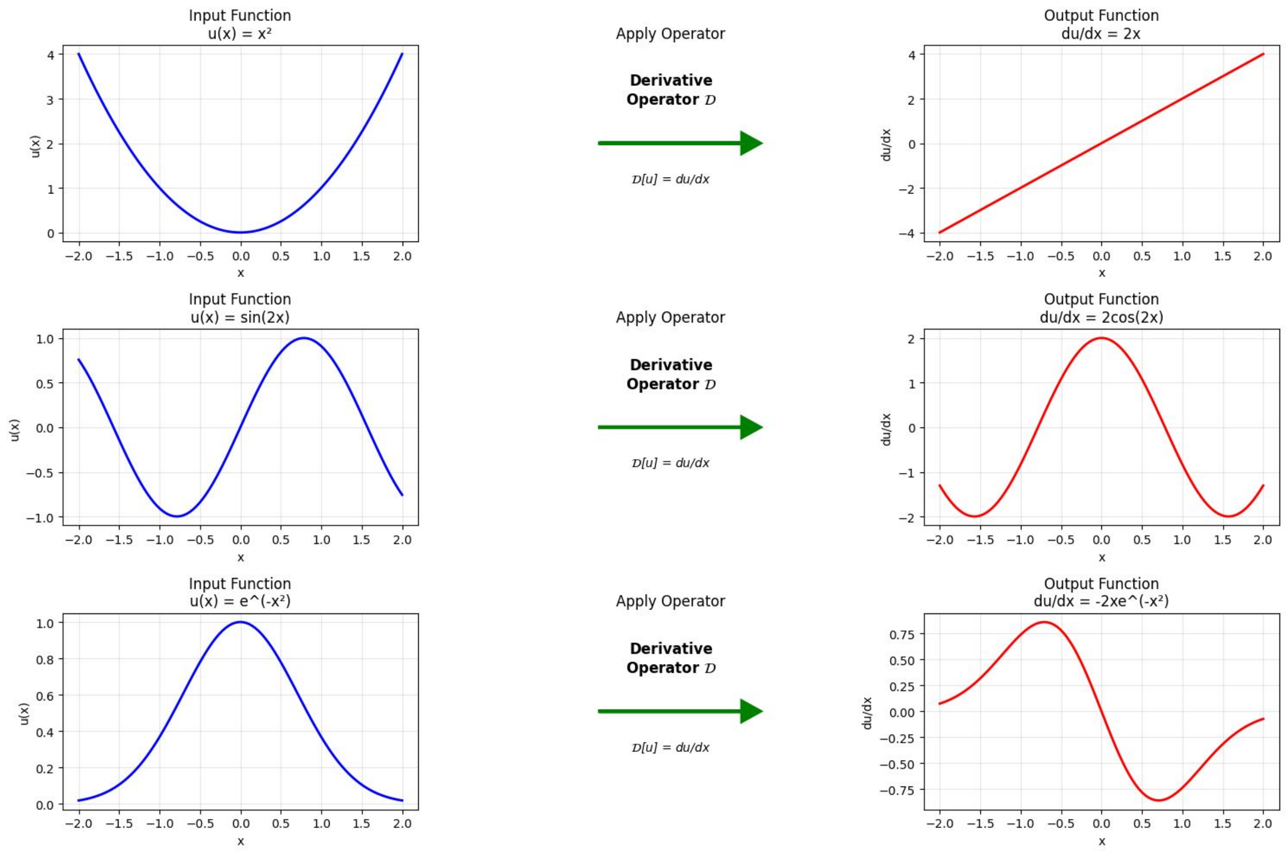
Function Encoder & Basis to Basis Operator Learning

Krishna Kumar

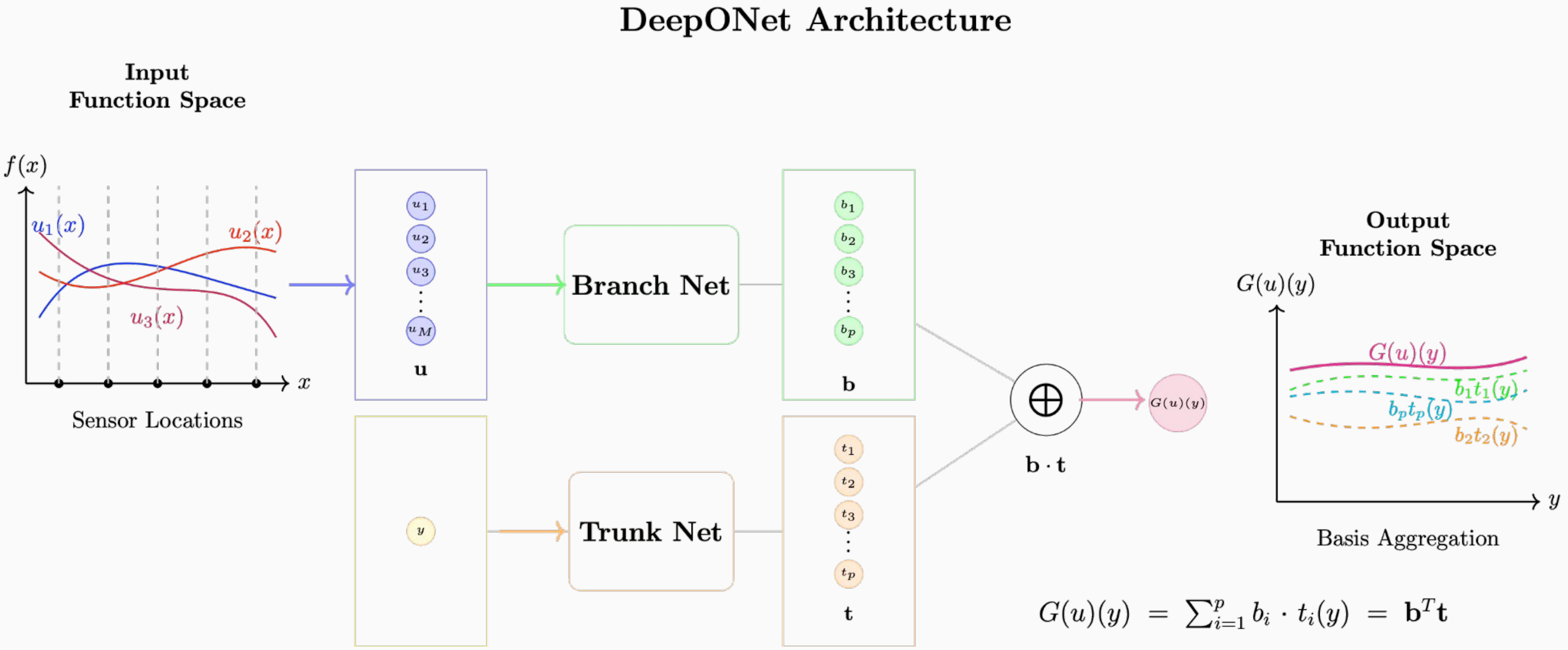
Learning Functions



Learning Operators



Operator Learning: DeepONet

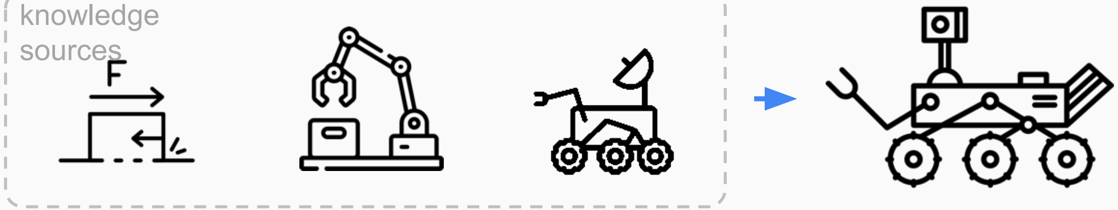


$$G(u)(y) = \sum_{i=1}^{p} b_i \cdot t_i(y) = \mathbf{b}^T \mathbf{t}$$

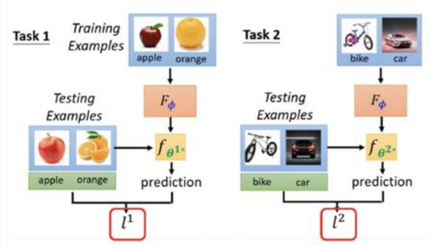
Zero-Shot: Learning to adapt or transfer

Adaptation updating or refining learned models using new data

Transfer leveraging knowledge from diverse sources

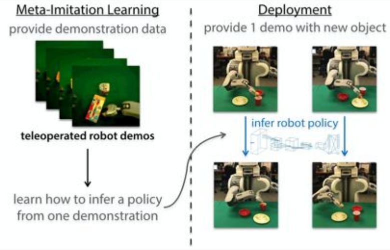


Existing transfer approaches



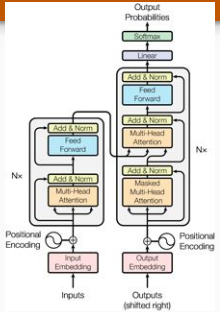
Meta-Learning

Chelsea Finn, Pieter Abbeel, Sergey Levine. (2017). Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks.



Imitation Learning

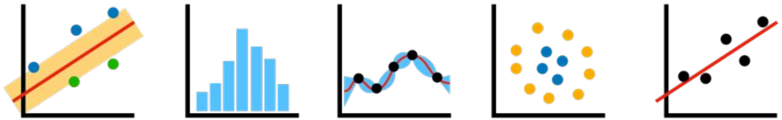
O'Neill, A., Rehman, A., Maddukuri, A., Gupta, A., Padalkar, A., Lee, A., ... & Chen, M. (2024). Open X-Embodiment: Robotic Learning Datasets and RT-X Models



Transformers

Ashish Vaswani, et. al. (2017). Attention is All you Need.

D. Celestini, D. Gammelli, T. Guffanti, S. D'Amico, E. Capello and M. Pavone. (2024). Transformer-Based Model Predictive Control: Trajectory Optimization via Sequence Modeling



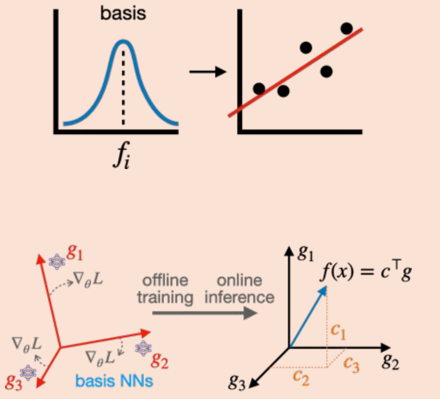
SVMs | Density Estimation | GPs | Kernel PCA | Ridge Regression

Hilbert Space Representations

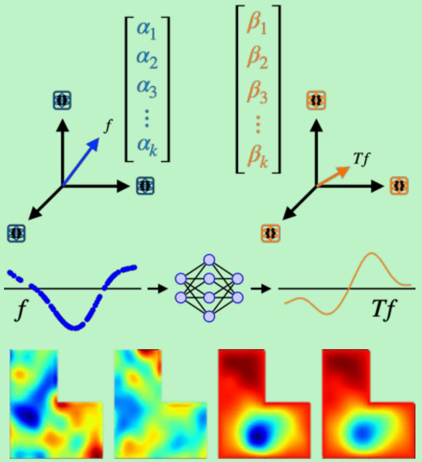
✅ guarantees ✅ interpretable ✅ efficient

Basis to Basis Operator Learning

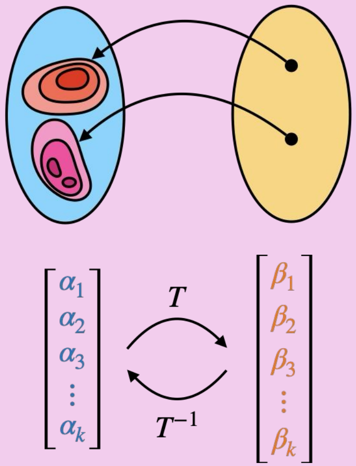
Function Encoders



Basis to Basis Operator Learning

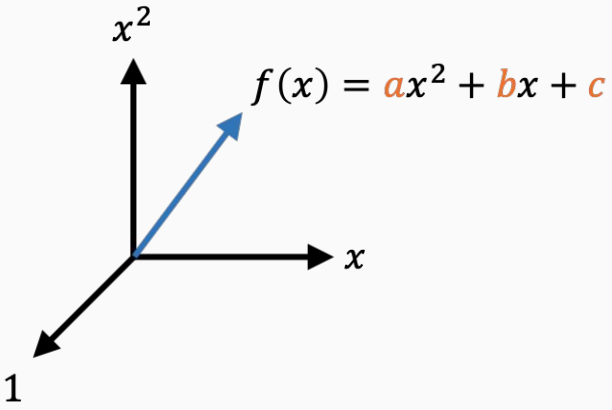


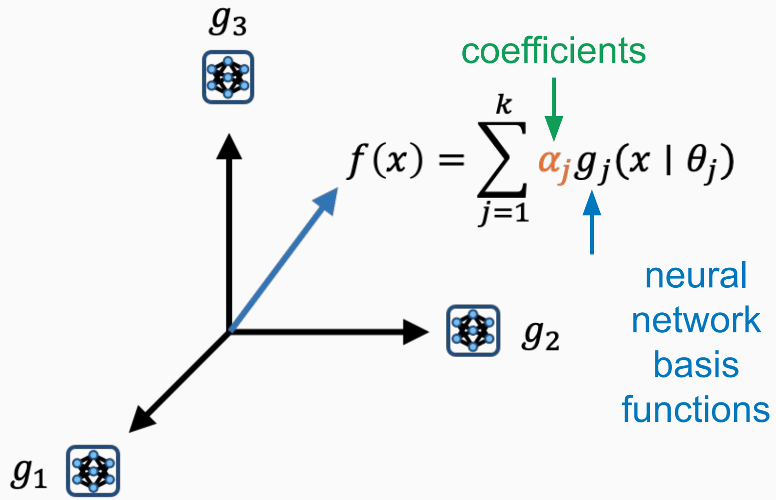
Inverse Neural Operators



Function encoders: combining neural networks and Hilbert spaces

Problem: How can we represent Hilbert spaces?





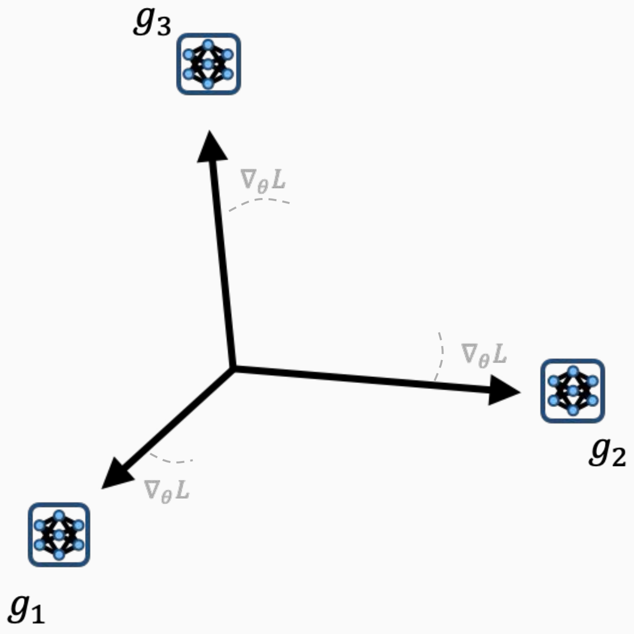
	simple polynomial example	**function encoders**
Basis	$\{1 \quad x \quad x^2\}$	$\{g_1 \quad g_2 \quad g_3 \quad \cdots \quad g_k\}$
Representation:	$[a \quad b \quad c]$	$[\alpha_1 \quad \alpha_2 \quad \alpha_3 \quad \cdots \quad \alpha_k]$

Ingebrand, T., Thorpe, A. J., & Topcu, U. (2025). Function Encoders: A Principled Approach to Transfer Learning in Hilbert Spaces.

Function Encoders: **offline training, online inference**

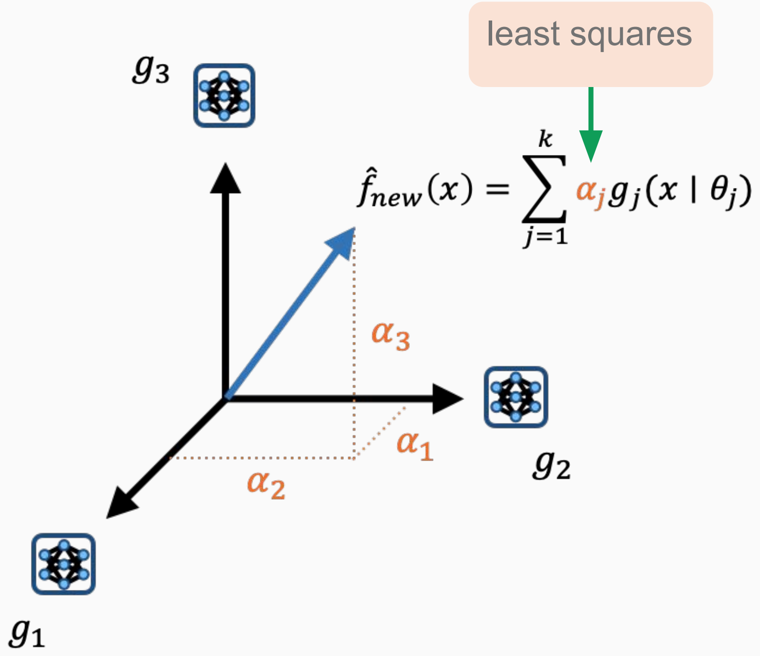
Offline Training

learn the basis functions



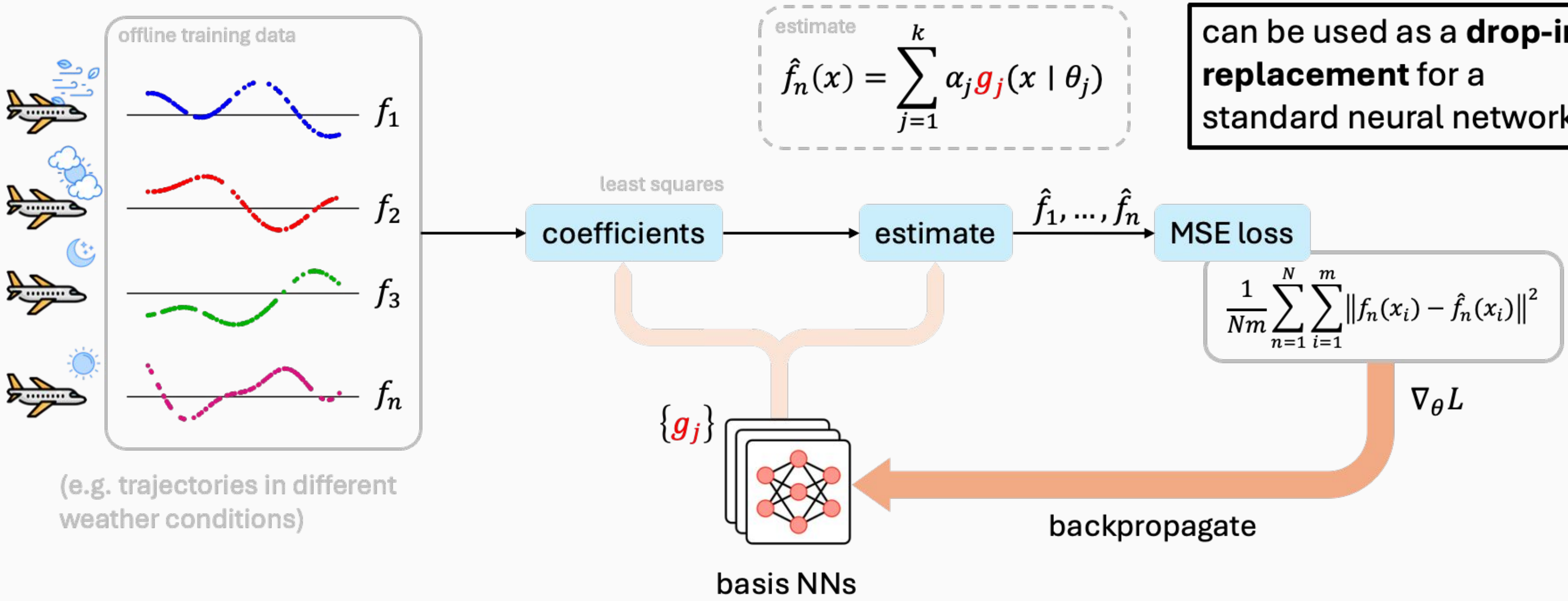
Online Inference

compute the coefficients α



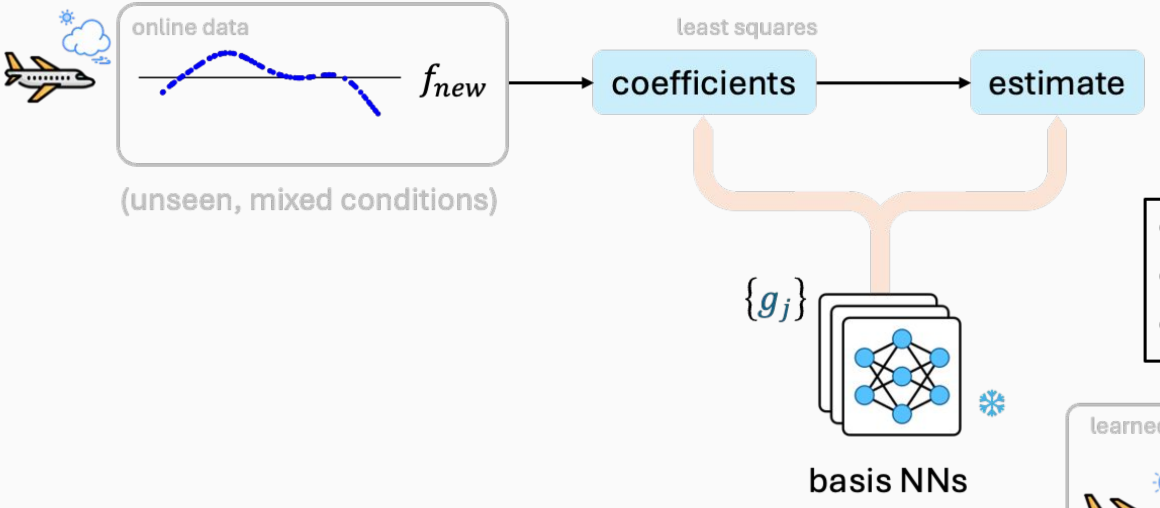
Ingebrand, T., Thorpe, A. J., & Topcu, U. (2025). Function Encoders: A Principled Approach to Transfer Learning in Hilbert Spaces.

Offline training



can be used as a **drop-in replacement** for a standard neural network

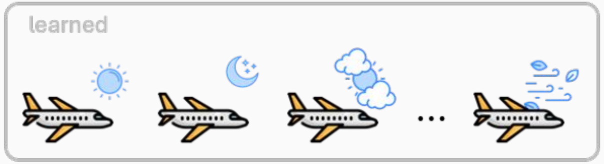
Online Inference



estimate

$$\hat{f}_{new}(x) = \sum_{j=1}^{k} \alpha_j g_j(x \mid \theta_j)$$

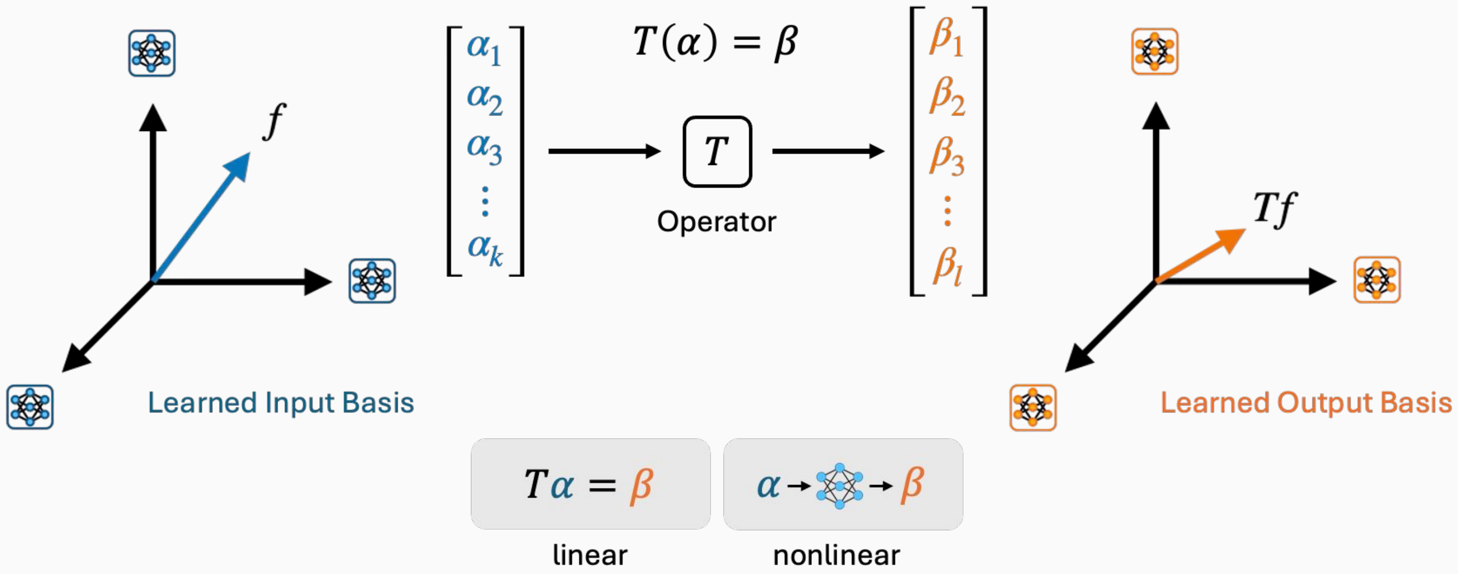
- **efficient** least squares computation
- **no network retraining** or fine-tuning
- **interpretable** via basis functions



Ingebrand, T., Thorpe, A. J., & Topcu, U. (2025). Function Encoders: A Principled Approach to Transfer Learning in Hilbert Spaces.

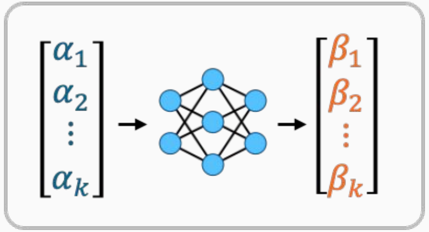
Basis-to-Basis (B2B) Operator Learning

Given: input-output pairs of transformations (f, Tf)
Goal: approximate $T: \mathcal{F} \to \mathcal{H}$



Ingebrand, T., Thorpe, A. J., Goswami, S., Kumar, K., & Topcu, U. (2025). Basis-to-basis operator learning using function encoders. CMAME

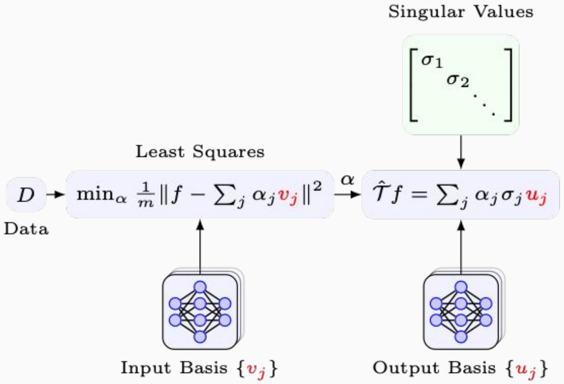
B2B Variants



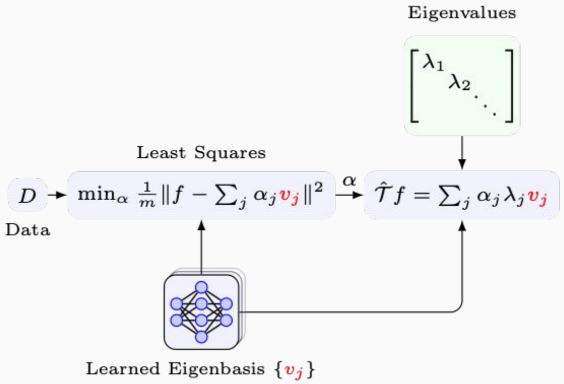
B2B (nonlinear)

$$\begin{bmatrix}\alpha_1\\ \alpha_2\\ \vdots\\ \alpha_k\end{bmatrix} \rightarrow \begin{bmatrix}T_{11} & T_{12} & \cdots & T_{1k}\\ T_{21} & T_{22} & \cdots & T_{2k}\\ \vdots & \vdots & \ddots & \vdots\\ T_{k1} & T_{k2} & \cdots & T_{kk}\end{bmatrix} \rightarrow \begin{bmatrix}\beta_1\\ \beta_2\\ \vdots\\ \beta_k\end{bmatrix}$$

B2B (linear)



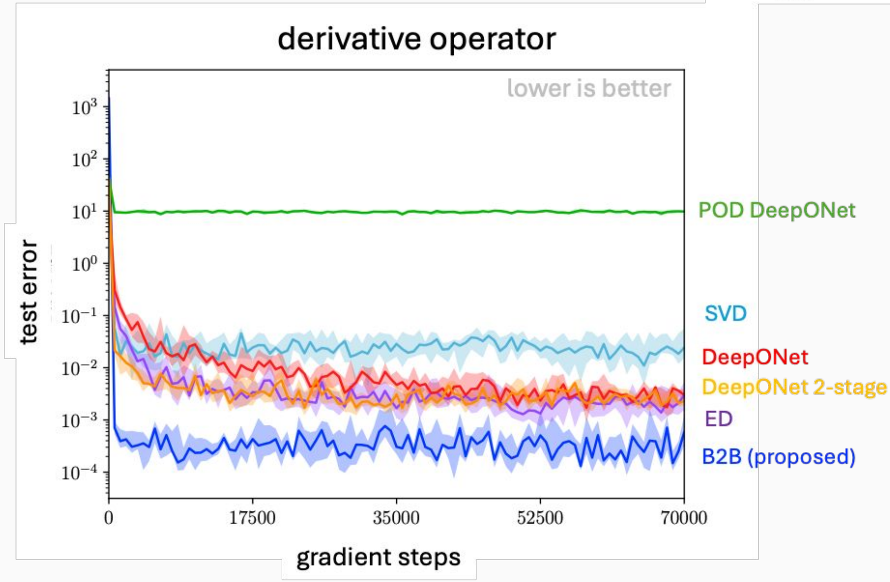
Singular Value Decomposition

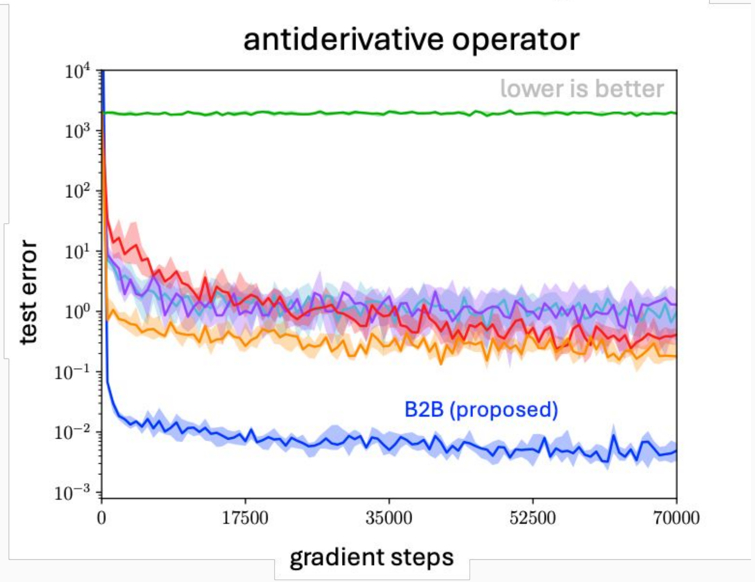


Eigen-decomposition

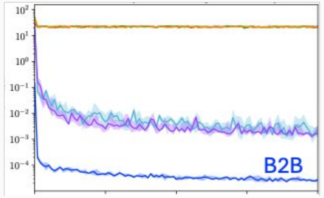
B2B linear example

$$\frac{ds(x)}{dx} = u(x), \qquad s(0) = 0, \qquad Tu(x) = s(x=0) + \int_0^x u(t)dt$$





varying sensor locations:



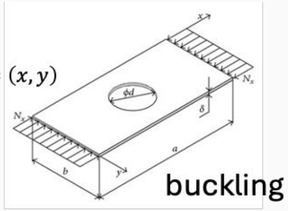
- **extrapolates** to the linear span
- **maintains accuracy,** even when the measurement locations **change**

B2B Elastic Plate

Modeling the solution of partial differential equations

input function

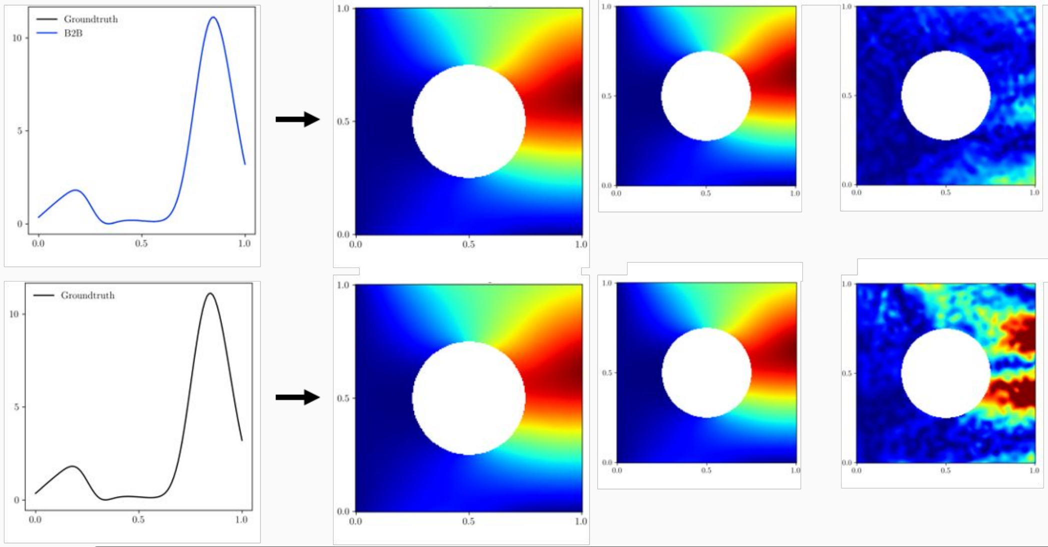
$$\nabla \cdot \sigma + f(\boldsymbol{x}) = 0, \boldsymbol{x} = (x, y)$$
$$(u, v) = 0, \forall x = 0$$



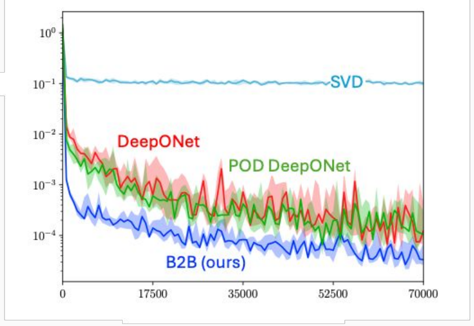
loading function (input function) | displacement (output function) | ground truth | error

B2B

DeepONet



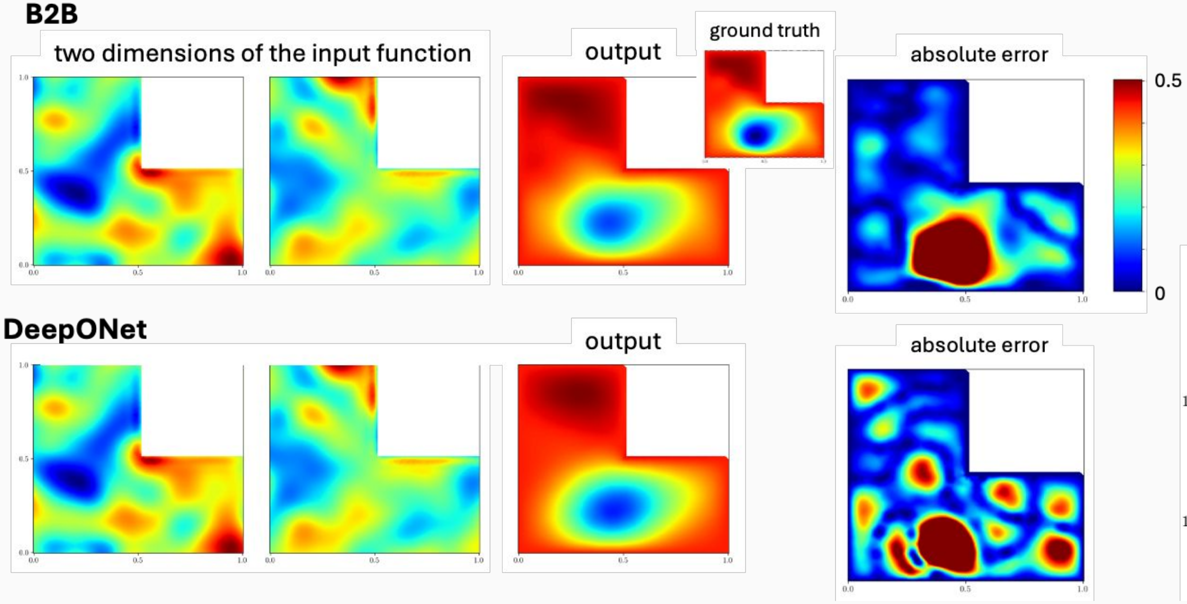
B2B has lower error, and **doesn't rely on a fixed grid or mesh.**

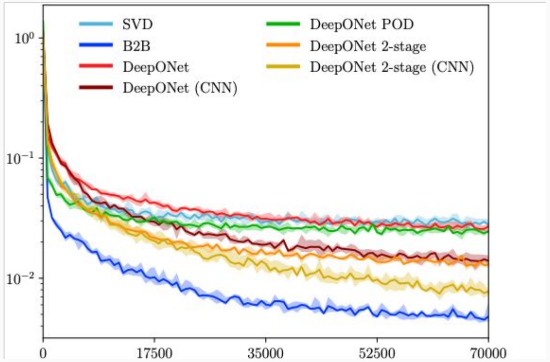


Neural operators model the **entire** solution, not just one instance!

B2B Darcy

$$\nabla \cdot \big(k(x)\nabla u(x)\big) + f(x) = 0, \quad x = (x,y) \in \Omega \coloneqq (0,1)^2 \times [0.5,1)^2,$$
$$u(x) = 0, \quad x \in \partial\Omega$$





B2B: Quantitative results

our proposed approaches

Dataset	Function encoders			DeepONet		
	B2B	SVD	Eigen	Vanilla	POD	Two-stage
Anti-derivative	**1.06e−02 ± 1.62e−02**	1.31e+00 ± 1.04e+00	2.02e+00 ± 2.63e+00	4.48e−01 ± 2.14e−01	1.96e+03 ± 1.34e+02	2.20e−01 ± 7.95e−02
Derivative	**8.63e−04 ± 6.60e−04**	3.33e−02 ± 2.03e−02	4.05e−03 ± 3.45e−03	3.68e−03 ± 2.57e−03	9.84e+00 ± 6.27e−01	2.33e−03 ± 1.01e−03
1D Darcy flow	**1.74e−05 ± 4.92e−06**	8.90e−04 ± 8.03e−05	–	4.47e−05 ± 8.94e−06	3.35e−05 ± 8.79e−06	2.59e−04 ± 8.43e−05
2D Darcy Flow	**5.30e−03 ± 1.19e−03**	2.89e−02 ± 2.31e−03	–	2.68e−02 ± 2.77e−03	2.50e−02 ± 1.64e−03	1.33e−02 ± 1.55e−03
Elastic plate	**6.30e−05 ± 5.59e−05**	1.03e−01 ± 1.83e−02	–	4.66e−04 ± 8.16e−04	5.59e−04 ± 1.15e−03	–
Parameterized heat equation	**4.07e−04 ± 2.86e−04**[a]	2.27e−01 ± 2.35e−02	–	6.00e−04 ± 1.09e−03	8.88e−01 ± 1.15e−01	–
Burger's equation	**5.07e−04 ± 1.93e−04**	1.01e−01 ± 1.16e−02	–	2.16e−03 ± 5.59e−04	1.94e+00 ± 1.76e−01	2.03e+00 ± 1.78e−01

[a] While the mean of prediction errors for B2B is lower than DeepONet for the parameterized heat equation dataset, the median is higher

B2B outperforms DeepONet on several PDE benchmarks